

Taller en Sala 5

Complejidad de Algoritmos



Objetivo: 1. Realizar estudios empíricos para validar una hipótesis sobre el comportamiento en tiempo de ejecución de un algoritmo con varios tamaños de problema 2. Usar la notación O para encontrar formalmente la complejidad asintótica en tiempo de algoritmos.



Consideraciones: Lean y verifiquen las consideraciones de entrega,



Trabajo en Parejas



Mañana, plazo de entrega



Docente entrega plantilla de código en GitHub



Sí .cpp, .py o .java



No .zip, .txt, html o .doc



Alumnos entregan código sin comprimir GitHub



En la carpeta Github del curso, hay un código iniciado y un código de pruebas (tests) que pueden explorar para solucionar los ejercicios



Estructura del documento: a) Datos de *vida real*, b) *Introducción* a un problema, c) Problema a resolver, d) Ayudas. Identifiquen esos elementos así:

a)



b)



c)



d)



PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

UNIVERSIDAD EAFIT

Acreditación Institucional
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

Ejercicios a resolver



En la vida real, el lenguaje de programación Python utiliza *Insertion sort* para ordenar arreglos de menos de 100 elementos y *Merge sort* para ordenar arreglos de más de 100 elementos.

1 La información financiera, de ventas, de mercadeo y de recursos humanos de las grandes empresas se almacena en grandes bases de datos. Para acceder a esa información de manera eficiente, la información tiene que estar ordenada, por algún criterio, por ejemplo, de menor a mayor.

El criterio para el ordenamiento se conoce como la llave única y es, por ejemplo, la cédula para una persona o el código del estudiante para un estudiante de Eafit. Un problema que existe es que el ordenamiento debe ser muy rápido para grandes volúmenes de datos.



Implementen el algoritmo de ordenamiento *insertion sort*.



Calculen la complejidad asintótica para el peor de los casos, tomen tiempos para 20 tamaños del problema diferentes, generen una gráfica y analicen los resultados. ¿Este algoritmo es apropiado para ordenar grandes volúmenes de datos?

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

**UNIVERSIDAD
EAFIT®**

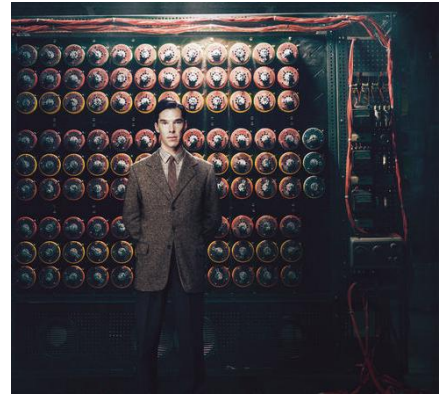
**Acreditación
Institucional**
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

ESTRUCTURA DE DATOS 1

Código ST0245

2 Alan Turing, antes de ser un héroe en la segunda guerra mundial, como es retratado en la película “El código enigma” (2014), había alcanzado gran renombre mundial por demostrar la equivalencia entre las funciones escritas de forma recursiva (el cálculo Lambda) y las funciones escritas con ciclos (la máquina de Turing).

Desde ese entonces, el problema que existe es comparar la eficiencia entre una solución hecha con ciclos y una con recursión

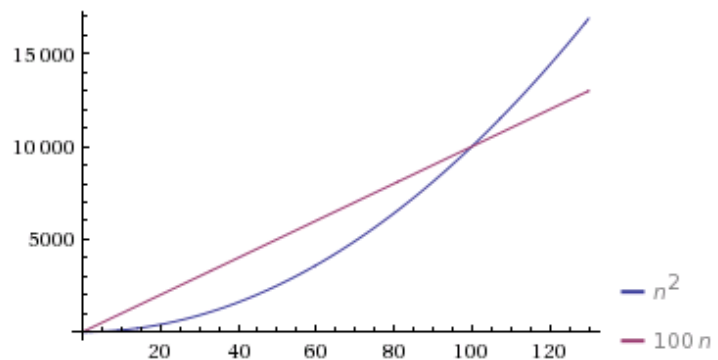


► **Implementen un algoritmo que sume los elementos de un arreglo utilizando ciclos**

► **Calculen la complejidad asintótica para el peor de los casos, tomen tiempos para 20 tamaños del problema diferentes, generen una gráfica y analicen los resultados.** ¿Existe una diferencia significativa, en el tiempo de ejecución, entre la implementación con recursión y la implementación con ciclos? ¿Por qué?

3 **[Ejercicio Opcional]** Históricamente, alrededor del 30% de los estudiantes que ingresan al curso de Estructuras de Datos y Algoritmos 1 confunde algoritmos con complejidad $O(n^2)$ con algoritmos $O(n)$.

Un algoritmo con complejidad $O(n^2)$ es el algoritmo que imprime en la pantalla las tablas de multiplicar



► **Implementen un algoritmo que calcule las tablas de multiplicar del 1 al n.**

► **Calculen la complejidad asintótica para el peor de los casos, tomen tiempos para 20 tamaños del problema diferentes, generen una gráfica y analicen los resultados.** ¿La teoría (notación asintótica) corresponde a lo encontrado de forma experimental al tomar los tiempos de la implementación?

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

UNIVERSIDAD
EAFIT

Acreditación
Institucional
Renovación
2018 - 2026
Resolución MEN 2158 de 2018

Ayudas para resolver los Ejercicios

Para todos los ejercicios..... **Pág. 5**



Para todos los ejercicios



Pista 1: Procedimiento sugerido

1. Implementen los algoritmos
2. Identifiquen qué representa el tamaño del problema para cada algoritmo
3. Identifiquen valores apropiados para tamaños del problema
4. Tomen los tiempos para los anteriores algoritmos con 20 tamaños del problema diferentes.
5. Hagan una gráfica en Excel para cada algoritmo y pasarla a Word luego
6. Entreguen el archivo de Word pasado a PDF. No entregar el código.



Pista 2: Vean la sección 4.2 de la *Guía de Laboratorios* muestra cómo generar arreglos con números aleatorios



Pista 3: Investiguen cómo generar un arreglo con número aleatorios de tamaño n .



Pista 4: Investiguen cómo tomar los tiempos de ejecución de un fragmento de código



Pista 5: Investiguen cómo graficar los tiempos de ejecución, usando, por ejemplo, Microsoft Excel



Pista 6: Vean en *Guía de Laboratorios*, Numeral 4.7, “Cómo calcular el tiempo que toma un código en ejecutarse en Java”.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



ESTRUCTURA DE DATOS 1
Código ST0245



Pista 7: Vean en *Guía de Laboratorios, Numeral 4.4*, “Cómo aumentar el tamaño del heap y del stack en Java”.



Pista 8: Vean en *Guía de Laboratorios, Numeral 4.5*, “Cómo visualizar el montículo (heap) y el stack, y el consumo total de memoria de Java”



Pista 9: Vean en *Guía de Laboratorios, Numeral 4.6*, *Cómo usar la escala logarítmica en Microsoft Excel 2013*.



Pista 10: Ejemplo de una tabla con tiempos en milisegundos:

	<i>N = 100.000</i>	<i>N = 1'000.000</i>	<i>N = 10'000.000</i>	<i>N = 100'000.000</i>
Suma de arreglo				
Tablas de multipl				
Insertion sort				

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez

Teléfono: (+57) (4) 261 95 00 Ext. 9473

Correo: mtorobe@eafit.edu.co

Oficina: 19- 627

Agenden una cita dando clic en la pestaña

-*Semana*- de <http://bit.ly/2gzVg10>